Panel: The security of downloadable executable content:
past, present and future

Aviel D. Rubin
rubin@bellcore.com
Bellcore
445 South St.
Morristown, NJ 07960

The emergence of the world wide web has enabled many new types of users and applications. One of the most useful features that has resulted from the existance of a ubiquitous browser is the ability to download executable content from remote sites. The functionality this enables is powerful.

Java, Javascript, and ActiveX are examples of executable content that can be downloaded to a machine running a browser. Java, in particular, can be used to run platform-independent applications which can be prototyped quickly. One of the touted features of the Java interpreter in Netscape's Navigator and Microsoft's Internet Explorer is that applets cannot leave the controled environment. Thus, there have been claims that Java is secure.

It has been shown that this is far from true. Felten and his team at Princeton [1] have broken the type system of Java. In addition, they have shown numerous system level flaws in several of the interpreters. The result is that Java applets can violate the security policies and can even run native code on the client machine.

David Hopwood at Oxford [3] has discovered several novel ways of breaking Java security as well. His two applet attack was used to run native code on a machine. The result is that the attacker who serves the applet gains complete control of the client host. Mark Ladue [5] has written several applets that demonstrate how other weaknesses of the Java interpret can be exploited.

ActiveX is different from the Java model. It assumes that content is digitally signed by trusted distributors. If the signature checks, the client blindly trusts the code, and no further security measures are taken. One problem with this model is that there is no global public key infrastructure to scale the solution to the web. It may be a reasonable technique for localized communities of interest.

Sun, Netscape, and Microsoft are working hard to revise the security model used to secure downloadable executable content. Javasoft recently hired a leading security professional, Li Gong, to oversee the process. Brewer and his team at Berkeley [2] and Jaeger et. al [4] have provided longer-term solutions that offer increased security. However, these solutions are either very platform dependent, or they rely on specialized hardware.

This panel is intended to discuss what has gone wrong in the past, what is being done to deal with this, and what long-term solutions might be applicable to secure downloadable executable content.

References

[1] Drew Dean, Edward W. Felten, and Dan S. Wallach. Java security: From hotjava to netscape and beyond. 1996 IEEE Symposium on Security and Privacy, pages 190-200, 1996. http://www.cs.princeton.edu/ ddean/java/.

[2] Ian Goldberg, David Wagner, Randi Thomas, and Eric A. Brewer. A secure environment for untrusted helper applications. USENIX Security Conference VI, pages 1{13, 1996.

[3] David Hopwood. http://ferret.lmh.ox.ac.uk/ david/java/, 1996.

[4] Trent Jaeger and Aviel D. Rubin. Building systems that flexibly download executable content. USENIX Security Conference VI, pages 131{148, 1996.

[5] Mark Ladue. http://www.math.gatech.edu/ mladue/hostileapplets.html, 1996.